Speech-Singing Discrimination using Geometric Methods

Yan Michalevsky*, Supervised by Ronen Talmon
Electrical Engineering Department, Technion, Israel Institute of Technology
Technion Campus, Haifa 32000, Israel
# ymcrcat@gmail.com

Abstract—Automatic audio classification is a growing area of interest applicable to media services, search engines and intelligent human-computer systems. Human utterance classification is a subset of audio signals classification. Tasks such as speaker verification and speech recognition are an old problem. But human utterance also includes singing, shouting and other forms involving human voice. Classification between these forms becomes an interesting and important challenge. We try to apply several geometric approaches with roots in spectral clustering and graphs theory as a data-driven alternative to the previous, model-based solutions. Our goal is to show the advantages of this approach over clustering and classification using traditional methods such as K-means in the original feature domain.

I. INTRODUCTION

Recently there has been a development in the area of multi-dimensional manifold learning and a computational framework called Diffusion Maps was developed [3]. This method is based on an interpretation of a certain version of spectral graph analysis as a diffusion process. Multiple application examples were shown, including discovery of intrinsic geometry of manifolds, image denoising and classification, and clustering. Application of Diffusion Maps to those problems yielded impressive results and gave us the motivation to try to apply it to audio signal classification. Another algorithm that was inspired by Diffusion Maps and the concept of Diffusion Distance of data points is Localized Diffusion Folders (LDF). We implemented this algorithm and tested its application to clustering audio data.

The report is organized as follows: in section II we present the Diffusion Maps framework and the LDF algorithm. In sections III and IV we cover the specifics of our implementation and present the results of clustering and classification of speech and singing data using the Diffusion Embedding. We continue with describing the evaluation of the Localized Diffusion Folders algorithm and show the results of applying it to our data. Finally there is a summary and presentation of our conclusions.

II. SCIENTIFIC BACKGROUND

A. Diffusion maps framework

When classifying our data to two groups we assume that the data is sampled from a low dimensional manifold, lying in a multidimensional space (N-dimensional, where N is the number of features we use to represent an audio sample). To simplify the problem we want to perform dimensionality reduction, and we can achieve it by capturing the underlying geometric structure of the data. Linear mapping is not applicable since the structure of the manifold is not necessarily linear. An example of such data is shown in Fig. 1. The points on the right are actually taken from a two-dimensional manifold (the Swiss-roll) lying in a three-dimensional space. Since the manifold is not linear we cannot apply a linear method, like PCA, to recover its structure. As we can see in section IV-B we manage to map that nonlinear manifold using Diffusion Maps.

Non-linear manifold learning methods include:
- ISOMAP [13]
- Locally Linear Embedding (LLE) [12]
- Laplacian Eigenmaps [1]
- Hessian Eigenmaps [5], and finally
- Diffusion Maps [3]

The basic idea behind those methods is to represent the data with distances instead of sample coordinates. A distance or affinity matrix is constructed according to some metric. In the case of Diffusion Maps we define a pairwise similarity matrix using a kernel of our choice. The specific choice of the kernel depends on the application and the data properties. A common choice is the Gaussian kernel according to which we get

\[ W_{ij} = K(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{\sigma^2}} = e^{-\frac{||x_i - x_j||^2}{2\epsilon}} \]

where \( \epsilon \) plays the role of the diffusion distance, or in other words some measure for radius we look in for neighbor points. \( W \) thus represents a fully connected graph where the data samples are the nodes and the transition probabilities are proportional to the corresponding elements of \( W \). To transform it to a Markov transition matrix we normalize each row by the sum of its elements (so the sum of probabilities of all
transitions is 1) and obtain the random walk matrix $M$

$$D_{ii} = \sum_j W_{ij}$$

$$M = D^{-1}W$$

So that

$$Pr(x(t+1) = x_j | x(t) = x_i) = M_{ij} = \frac{W_{ij}}{\sum_j W_{ij}}$$

To run the diffusion process we can simply perform the random walk by raising the matrix $M$ by power of $t$ to take us $t$ steps forward [3].

For large enough values of $\epsilon$ the graph is fully connected so that $M$ represents an irreducible and aperiodic Markov chain. Consequently it has a unique eigenvalue equal to 1. There are different ways to derive an embedding from the Diffusion Maps framework. One is by defining the Graph Laplacian [1] as $L = D - W$ and given an arbitrary embedding

$$\frac{1}{2} \sum_{i,j} W_{ij}||y_i - y_j||^2 = Tr(Y^TLY)$$

to find the optimal $Y$ that preserves locality. The solution is given by the eigen-decomposition of the normalized Graph Laplacian matrix and is related to the Markov transition matrix $M$ by

$$L\psi = \lambda D\psi \iff M\psi = (1 - \lambda)\psi$$

Another way is to define a Diffusion Distance as in [11] and show that it is equivalent to the Euclidean distance in the Diffusion Map space. Using a few largest eigenvectors (excluding the eigenvector that equals to 1) for approximation of the Diffusion Distance results in dimensionality reduction of our data.

The Diffusion Maps embedding into an n-dimensional space is defined by

$$\Psi_t(x_i) \triangleq \begin{bmatrix} \lambda_1^t \psi_1(x_i) \\ \lambda_2^t \psi_2(x_i) \\ \vdots \\ \lambda_n^t \psi_n(x_i) \end{bmatrix}$$

An example of capturing the intrinsic geometry using Diffusion Maps is shown in Fig. 2. The input data was generated by pictures of the text “3D” rotated in different angles by controlling two parameters, $\alpha$ and $\beta$, corresponding to rotation around the vertical and the horizontal axes respectively. Using Diffusion Maps the authors mapped the input data into $\mathbb{R}^2$ using the first two nontrivial eigenvectors, and were able to capture those rotational degrees of freedom and produce a surface that could have been the source of such data [10].

### B. Choice of the scale parameter

Careful choice of the scale parameter $\epsilon$ is crucial to obtain a good embedding. The order of $\epsilon$ is what usually important and not its exact value. In physical terms we want to ensure that some diffusion takes place but on a small enough scale, to recover the local geometry. Two methods are suggested in [8]. The first is the Singer Measure that involves calculation of the volume of the distance matrix $W$ for different values of $\epsilon$

$$SM(\epsilon) = \sum_{i,j} e^{-\frac{\|x_i - x_j\|^2}{\epsilon}}$$

and choosing a value from the range where the log-log plot of $SM(\epsilon)$ is linear. This indicates that diffusion occurs, but does not reach saturation which results in collapsing of all the points of the diffusion map plot to a very small ball.

According to the second approach, denoted as the max-min measure, $\epsilon$ is chosen in a way that the kernel describes infinitesimal connectivity of the data set. Simply speaking, we want to choose the smallest $\epsilon$ that still maintains local connectivity. The minimal distance that connects the point $i$ to its neighborhood is

$$H_i(\epsilon) = \min_j (||x_i - x_j||^2), i \neq j$$

and we choose

$$\epsilon = C \cdot \max(H(\epsilon))$$

(where $C$ is a constant in the range of 2-3) to ensure that each point in the graph is connected.

### C. Geometric Harmonics

Using the previously presented embedding method it is possible to analyze the intrinsic geometry of the data and use it for clustering in the Diffusion space. However it is sometimes necessary not only to represent a given set of samples but to be able to efficiently extend this representation to new samples. In our case we would like, given an embedding of the training set of speech and singing samples, to extend it to new samples in order to be able to perform their classification in the Diffusion space.

**Geometric Harmonics**, introduced in [10], are based on the Nyström extension:

$$\Psi_j(x) = \frac{1}{\lambda_j} \int_X p(x,y) \psi_j(y) p(y) dy, \forall x \in X$$

For all $x$-s in the training set the functions $\Psi_j$ and $\psi_j$ agree. For $x$-s outside the training set $\Psi_j$ is an extension of $\psi_j$. It is extended as an average of its values on the training set. Practically, given the discrete training set represented by the matrix $K = [x_1, ..., x_M]^T$ with the spectral representation $\{\lambda, \psi\}_j^{M}$ we extend it to the new sample $x_{M+1}$ by

$$[\psi_j]_{M+1} = \frac{1}{\lambda_j} \sum_{i=1}^M \frac{K(x_i, x_{M+1})}{D} [\psi_j]_i$$

where $K(\cdot, \cdot)$ is the kernel we applied to calculate the distance matrix $W$, and

$$D = \sum_{i=1}^M K(x_i, x_{M+1})$$

In our case, when we have to classify a new sample we find its embedding coordinates using the algorithm described above and then we can apply k-NN to associate it with a certain group according to its nearest neighbors in the Diffusion space.
Figure 2. Diffusion Maps capture the intrinsic parameters of the data.

D. Localized diffusion folders

Localized Diffusion Folders is a clustering algorithm developed by G. David, A. Averbuch and R. Coifman [4]. Its purpose is hierarchical clustering of high-dimensional data. Diffusion Folders are data partitioning into local neighborhoods, based on a diffusion graph. This algorithm captures the local geometry and adapts to local sample density. It is also good for de-noising and elimination of erroneous connections between unrelated points in the graph, created by diffusion through noisy samples. The algorithm consists of the following main steps:

**Algorithm 1 LDF algorithm**

2) Construction of a similarity matrix using some metric (Euclidean, Weighted Euclidean, Cosine or Mahalanobis Distance metric).
3) Application of an adaptive Gaussian kernel to the affinity matrix with automatic determination of the scale control for each point.
4) Initial partitioning into non-overlapping neighborhoods, and then reassignment of each point to its nearest neighborhood according to the centroid of each one. The result is a Voronoi diagram that depends on the random choice of points during the initial partitioning.
5) Several Voronoi diagrams are constructed independently according to the previous steps, and all are fused into a single partitioning to eliminate noise.
6) Upper levels are constructed by calculating the affinity between different folders and using the localized affinity matrix as an input to another iteration of the algorithm in a recursive way.

III. SPEECH-SINGING DISCRIMINATION

In our work we tested both clustering and classification. By clustering we mean automatic separation of samples into two distinct groups and comparison of the result with the real labels of the samples. Classification aims at labeling an unlabeled sample, based on a previously processed training set. Our classification system is schematically shown in Fig. III. Here we cover the different steps of the classification algorithm.

After initial normalization and preprocessing (explained in III-A) we compose high-dimensional feature vectors using various features extracted from the audio samples. The features are described in III-B. Then we apply the algorithm described in II-A. Each feature vector represents a sample of a high-dimensional manifold.

Since the characteristic scale of different features is non-homogenous we use the Gaussian kernel with a certain modification. Instead of using a uniform scale control \( \epsilon \) for all features, i.e. for each element of the feature vector, we use a different value for each element. The kernel now has the following form

\[
K(x_i, x_j) = e^{-\frac{1}{2}(x_i - x_j)^T \Sigma^{-1} (x_i - x_j)}
\]

where

\[
\Sigma_{M \times M} = \begin{pmatrix}
\epsilon_1 & \cdots & 0 \\
\vdots & \ddots & \vdots \\
0 & \cdots & \epsilon_M
\end{pmatrix}
\]

and \( M \) is the number of features. \( \{\epsilon\}_{i=1}^{M} \) are chosen as the variance of the \( i \)-th feature across all samples, multiplied by some constant \( C \) which depends on the data, and its value is chosen according to an optimization performed during the training stage. Using a different scale for each feature takes into account its range of values which might differ from that of other features.

We apply the modified Gaussian kernel to each pair of vectors to calculate the distance matrix \( W \). After normalizing its rows we get a Markov transition Matrix \( M \). Its spectral representation is obtained using Eigen-decomposition and we use the eigenvalues \( \{\lambda\}_{i=1}^{N} \) and eigenvectors \( \{\psi_{N \times 1}\}_{j=1}^{M} \) to calculate the Diffusion embedding of the original feature vectors. This embedding serves as a reference for classification of new samples. For each new sample we apply the same normalization and preprocessing steps, and extract the same features as for the training set. We acquire a new feature vector. Applying the Gaussian kernel to the new vector with each vector in the training set we calculate the Geometric-Harmonics and are able to extend our Diffusion-map to the new sample. Having found the approximate embedding coordinates of the new sample in the Diffusion space we use K-NN to classify it according to its nearest neighbors from the training set, and label the new sample as Speech or Singing.

A. Preprocessing

We perform a simple preprocessing of the samples prior to performing feature extraction. All audio files are resampled to
a sampling rate of 16 KHz so that the features extracted from each file would be comparable. This sampling rate is high enough to capture human voice frequencies. We also split the files to segments of 4 seconds each (2 second segments were tried as well) to obtain more sample points. Increasing the number of samples is important to get a smoother manifold which is important to get a more accurate extension using Geometric Harmonics, and a better picture of the manifold intrinsic geometry. Also, prior to feature extraction, the audio samples are normalized to prevent the calculated spectral energy from being dependent upon the audio volume.

**B. Features**

We focus on the problem of discrimination between speech and singing. Such capability is applicable, for example, to filtering radio streaming, which has periods of talking and singing. Also, it could be used to search and sort big collections of audio data. This is an nontrivial task since the computationally measurable differences between speech and singing are not very clear. Several works refer to various features that are helpful to distinguish those to forms of utterances [2], [6], [7], [9]. Features such as pitch track statistics (also referred fundamental frequency or $f_0$), vibrato measure, Mel Frequency Cepstral Coefficients are mentioned in the literature. Vibrato, defined in some articles as a 4-Hz modulation of the pitch track, is considered to be a good indicator of singing [2], [7], [9]. Another useful feature is the percentage of voiced and unvoiced frames. In singing there are in general more voiced frames than in plain speech. A wide range and big variance of pitch values might be good indicators of singing as well. Silence can also be a cue - songs tend to have less silent periods than talking records. In general, the difference between speech and singing lies in the long-term rather than in the short-term features [2]. Locally, there might appear some big jumps in the pitch track, or a lot of voiced frames. Therefore these properties have to be measured over a long enough period of time.

We tried however to work without making too many assumptions on the properties of the sample and see whether geometric methods can recover the subtle differences between singing and speech. Therefore we have chosen to use the following features for the final evaluation:

1) Mel Frequency Cepstral Coefficients (MFCC).

2) Pitch statistics (mean, variance, minimum and maximum) and pitch derivative statistics.

3) Voiced/Unvoiced frames percent.

We tried using both the combination of all features together and each one separately. Other features that were omitted in the final evaluation included:

1) Zero-crossing rate.

2) Short-time energy.

**IV. Evaluation**

**A. Using Diffusion Maps**

We evaluated the algorithm with two different collections of samples. In the first one, the singing samples were taken from a capella tracks of real songs or audio samples for electronic music composition. In those samples it is clearly heard that they belong to the category of singing. The speech samples were taken from the TIMIT database, which is clear speech. Using Diffusion Maps technique with all mentioned features, we embedded the corpus samples into the Diffusion space and performed clustering using K-means and classification of test samples using K-nn. Clustering in the original feature space yielded an error of 12.4% and in the Diffusion space an error of 6.7%. Classification using K-nn in the original feature space yielded an error of 2.8% and using Geometric Harmonics in the Diffusion space an error of 1.4%, meaning the embedding improved the classification. The three dimensional embedding using the first eigenvectors (not including the one corresponding to eigenvalue 1) is shown in Fig. IV-A. Using MFCC only we obtained an error of 13.4% in the original features space, and 6.5% in the Diffusion space for clustering. For classification we obtained an error of 3.8% in the original space and 2.4% in the Diffusion space. The embedding results are presented in Fig. IV-A.

Using MFCC only we obtained an error of 13.4% in the original features space, and 6.5% in the Diffusion space for clustering. For classification we obtained an error of 3.8% in the original space and 2.4% in the Diffusion space. The embedding results are presented in Fig. IV-A.

Despite getting quite good results we didn’t get an obvious improvement of clustering and classification using Diffusion Maps over K-means and K-nn applied to the original feature vectors. We therefore wanted to test our method on another corpus. We also wanted to work with more homogeneous samples where it was less obvious whether it is speech or singing. D. Gerhard collected such a corpus, recording different speakers for his Ph.D. research [7]. In this corpus several people both pronounce (speech) and sing different
phrases. In Fig. 6 we show the embedding of MFCC and MFCC-delta features calculated over Gerhard’s corpus using two methods: Diffusion Maps and Isomap. The clustering error using Diffusion Maps was 18.9% and 46% using Isomap embedding. With K-means an error of 46.7% is obtained for the same features. Classification by K-NN in the original feature space yields an error of 18%. We see that using an embedding gives a great improvement for both clustering and classification. Isomap supposedly fails in this case because it does not use a different scale for each feature.

Fig. 7 demonstrates the embedding of feature vectors based solely on the pitch track statistics. As we can see MFCC and MFCC-delta features give a better separation of speech and singing samples using Diffusion Maps. Classification by K-NN yielded an error of 34.2%, while an error of 26.5% was obtained using Diffusion Maps.

B. Application of LDF

Looking for a better clustering of our data we implemented the LDF algorithm. We first evaluated the algorithm on synthetic 3-D manifolds such as the Swiss Roll and the S-shape to test it. Using LDF requires a careful calibration of the initial scale and a criterion for choosing points for the initial partitioning to local neighborhoods. These steps are not described thoroughly in [4]. In Fig. 8 we show the result of clustering a Swiss-roll using our implementation of LDF. As we can see there are several incorrect associations of points lying in different areas of Swiss-roll to the same folder. A possible cause for these errors is that the input data was not dense enough, or an inadequate implementation of the two mentioned steps of choosing the initial scale and the initial partition.

For comparison, we present in Fig. 9 the mapping of a Swiss-roll using four different methods: K-means, LLE, Isomap and Diffusion Map. Diffusion Maps and Isomap provided the best mapping of the different areas of the shape, when in this particular execution Diffusion Maps gave a better result than Isomap. That might not be the result of a better embedding but of the application of K-means on the embedded data.

We applied LDF clustering to feature vectors based on the pitch track statistics of our audio samples. In Fig. 10 we present the result in 3 dimensions out of all 6 we actually used for the feature vectors. As we can see on the left graph there is no good enough separation between speech and singing and therefore we cannot have any magic using LDF either.

V. SUMMARY AND CONCLUSIONS

In this work we applied recently developed dimensionality reduction and clustering methods to auditory data. We tried to utilize relatively raw features and rely on the learning algorithms to provide a good classification. We have demonstrated an improvement with Diffusion Maps over K-means in the original features space, for both corpuses. Even when the discrimination in the original feature space was good, the application of Diffusion Maps embedding had a noticeable effect. And when the separation between speech and singing was not clear in the original features space (D. Gerhard’s corpus) we were able to improve it significantly by embedding. Possibly, the temporal development and not the spectral information could give more insight into the problem of discrimination between speech and singing. Although Cepstral Coefficients and Pitch Track statistics do contain information about the temporal development it is not enough to obtain a very good classification in those hard cases we face in D. Gerhard’s
Figure 6. Diffusion Map and Isomap embedding of MFCC and MFCC-delta features calculated on D. Gerhard’s samples. Clustering error using Diffusion embedding was 18.9% and 46% using Isomap embedding.

Figure 7. Embedding of feature vectors based on Pitch track statistics only. Clustering using Diffusion Maps yielded an error of 35.8% and 38.7% using Isomap.

Figure 9. Swiss-roll mapping using K-means, LLE, Isomap and Diffusion Map.
sample collection. We assume that a better utilization of temporal features and a better understanding of how to handle time series in the context of Diffusion Maps, may improve the results.

REFERENCES


